

PARALLEL SIMULATION OF BEAM-BEAM INTERACTION IN HIGH ENERGY ACCELERATORS*

Ji Qiang[†], Miguel A. Furman, and Robert D. Ryne, Lawrence Berkeley National Laboratory
Berkeley, CA 94720

Abstract

In this paper, we present a self-consistent simulation model of beam-beam interaction in high energy accelerators. Using a parallel particle-in-cell approach, we have calculated the electromagnetic fields between two colliding beams. Dynamic load balance is implemented to improve the parallel efficiency. A preliminary performance test on IBM SP Power3, Cray T3E and PC cluster is presented. As an application, we studied the coherent beam-beam oscillation in the proposed Large Hadron Collider.

1 INTRODUCTION

High energy accelerators are essential to study the inner structure of nuclear and elementary particles in modern physics. Inside a high energy accelerator, two charged particle beams moving with speed close to the speed of light in opposite directions collide at interaction point. High luminosity of two colliding beam is required in order to provide detailed understanding of physical phenomena such as CP-violation. However, the electromagnetic interaction between two charged beams, i.e. beam-beam interaction, will put a strong limit on the luminosity in the high energy accelerators. An accurate simulation of the beam-beam interaction will help to optimize the luminosity in high energy accelerators.

Beam-beam interaction has been studied in the past 30 years. Due to the typical computation cost required to accurately simulate the beam-beam interaction for many turns (10^4 to 10^6 turns), most of the previous studies used either a simplified physical model, e.g. neglecting the electromagnetic force of one beam [1, 2, 3, 4], or an approximated computation model, e.g. assuming the Gaussian shape of particle distribution or a pancake model [5, 6, 7, 8, 9, 10, ?]. To study the beam-beam interaction fully self-consistently for both beams, and to include the physical processes such as long range off-centroid interaction, the finite beam bunch length effects, and crossing angle collision, will require the computation resource beyond the capability of current serial computer. As far as we know, there is no existing code that can simultaneously handle these physical processes accurately. In this paper, we present a parallel beam-beam simulation model which can study these physical processes accurately by using high performance computers.

The organization of the paper is the following: The physical mode and computation methods are described in Sec-

tion 2. The parallel implementation is given in Section 3. An application to the study of coherent beam-beam oscillation in Large Hardon Collider (LHC) is given in Section 4. We summarize our preliminary results in Section 5.

2 PHYSICAL MODEL AND COMPUTATION METHODS

In the high energy accelerators, each charged particle has a position in phase space with coordinate $(x, x', y, y', \Delta z/\sigma_z, \Delta p_z/\sigma_{p_z})$. Here, superscript prime denotes $\partial/\partial s$ and s is the longitudinal position. The motion of the particle will be subject to the influence of external field, which provides the transverse and longitudinal focusing of the beam. The particle will also lose its energy through synchrotron radiation. This leads a process called radiation damping and quantum excitation, which will affect the particle motion inside accelerators. The Coulomb interaction among charged particles within a bunch is negligible due to the cancellation of electric force and magnetic force at high speed. However, for the opposite moving charged particles, the electric force and magnetic force will add up. This force can significantly affect the motion of the charged particles in the other beam.

The effects of external field can be represented, in the small-amplitude approximation, by a one-turn linear map, i.e.

$$\begin{aligned} x_{n+1} &= (\cos(2\pi\nu_{0x}) + \alpha_x \sin(2\pi\nu_{0x})) x_n \\ &\quad + \beta_x \sin(2\pi\nu_{0x}) x'_n \end{aligned} \quad (1)$$

$$\begin{aligned} x'_{n+1} &= -\gamma_x \sin(2\pi\nu_{0x}) x_n \\ &\quad + (\cos(2\pi\nu_{0x}) - \alpha_x \sin(2\pi\nu_{0x})) x'_{n+1} \end{aligned} \quad (2)$$

where α_x, β_x and γ_x are lattice functions at the interaction point, and ν_{0x} is horizontal lattice tune. A similar map applies to the vertical phase space y and y' by replacing $x \rightarrow y$ in above equation. For the longitudinal phase space, the one-turn map is defined by

$$\begin{aligned} \begin{pmatrix} \Delta z/\sigma_z \\ \Delta p_z/\sigma_{p_z} \end{pmatrix}_{n+1} &= \begin{pmatrix} \cos(2\pi\nu_s) & \sin(2\pi\nu_s) \\ -\sin(2\pi\nu_s) & \cos(2\pi\nu_s) \end{pmatrix} \\ &\quad \times \begin{pmatrix} \Delta z/\sigma_z \\ \Delta p_z/\sigma_{p_z} \end{pmatrix}_n \end{aligned} \quad (3)$$

where ν_s is the synchrotron tune.

The effects of radiation damping and quantum excitation can be represented using a localized stochastic map. For each particle, the map consists of the following transformations [5]:

$$x_{n+1} = \lambda_x x_n + r_1 \sigma_x \sqrt{1 - \lambda_x^2} \quad (4)$$

* Work supported in part by DOE Advanced Computing for 21st Century Accelerator Science and Technology Project.

[†] Email: jiqiang@lbl.gov

$$x'_{n+1} = \lambda_x x'_n + r_2 \sigma_{x'} \sqrt{1 - \lambda_x^2} \quad (5)$$

$$y_{n+1} = \lambda_y y_n + r_3 \sigma_y \sqrt{1 - \lambda_y^2} \quad (6)$$

$$y'_{n+1} = \lambda_{y'} y'_n + r_4 \sigma_{y'} \sqrt{1 - \lambda_{y'}^2} \quad (7)$$

$$\Delta z_{n+1} = \lambda_z \Delta z_n + r_5 \sigma_z \sqrt{1 - \lambda_z^2} \quad (8)$$

$$\Delta p_{z_{n+1}} = \lambda_{p_z} \Delta p_{z_n} + r_6 \sigma_{p_z} \sqrt{1 - \lambda_{p_z}^2} \quad (9)$$

where the σ 's are the nominal rms equilibrium beam sizes in each dimension, the λ 's are given in terms of the damping time τ (measured in units of turns) by $\lambda_i = \exp(-1/\tau_i)$ where i denotes x , y , or z , and the r 's are independent random numbers satisfying

$$\langle r_i \rangle = 0 \quad (10)$$

$$\langle r_i r_j \rangle = \delta_{ij} \quad (11)$$

The first term in above transformation represents the radiation damping, and the second term represents the quantum excitation.

To calculate the electromagnetic force from beam-beam interaction, we have used a multiple slice model. In this model, each beam bunch is divided into a number of slices in the longitudinal direction. Each slice contains nearly the same number of particles at different longitudinal location z . The collision point between two opposite slice i and j is determined by

$$S_c = \frac{1}{2}(z_i^+ - z_j^-) \quad (12)$$

The transverse coordinates of the particles at collision point are given by

$$x^c = x + S_c x' \quad (13)$$

$$y^c = y + S_c y' \quad (14)$$

Then the slopes of the particles are updated using the beam-beam electromagnetic forces at the collision point following

$$x'_{new} = x' + \Delta x' \quad (15)$$

$$y'_{new} = y' + \Delta y' \quad (16)$$

where

$$\Delta x'_{2,1} = \frac{2q_1 q_2 N_{1,2}}{\gamma_{2,1} 4\pi\epsilon_0 m_{2,1} c^2} E_{x_{1,2}} \quad (17)$$

$$\Delta y'_{2,1} = \frac{2q_1 q_2 N_{1,2}}{\gamma_{2,1} 4\pi\epsilon_0 m_{2,1} c^2} E_{y_{1,2}} \quad (18)$$

where subscript 2, 1 represents contributions from beam 1 or beam 2, $\gamma = 1/\sqrt{1 - \beta^2}$, $\beta_i = v_i/c$, $i = x, y, z$, c is the speed of light, ϵ_0 is the vacuum permittivity, q is the charge of the particle, m is the rest mass of particle, N is the number of particles in a bunch, and E_x and E_y are the transverse electrical fields generated by the opposite moving beam. After the collision, the particles of each slice drift back to their original locations following

$$x = x^c - S_c x'_{new} \quad (19)$$

$$y = y^c - S_c y'_{new} \quad (20)$$

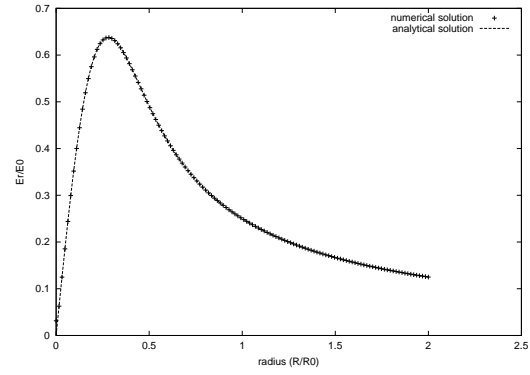


Figure 1: Radial electric field as a function of radius.

The electrical field generated by the opposite moving beam can be obtained from the solution of Poisson's equation. The solution of Poisson's equation can be written as

$$\phi(x, y) = \int G(x, \bar{x}, y, \bar{y}) \rho(\bar{x}, \bar{y}) d\bar{x} d\bar{y} \quad (21)$$

where G is the Green's function, and ρ is the charge density. For the case of transverse open boundary conditions, the Green's function is given by:

$$G(x, \bar{x}, y, \bar{y}) = -\frac{1}{2} \ln((x - \bar{x})^2 + (y - \bar{y})^2) \quad (22)$$

An FFT based method can be used to calculate the convolution efficiently [11]. In this algorithm, the particle domain and the field domain are in the same computation domain. In the beam-beam interaction, two opposite moving beams may not overlap with each other transversely. The field domain generated by one beam bunch could be different from the particle domain containing the beam. In the paper, we have extended above FFT based algorithm to handle this situation. The center of the field domain is shifted to the center of the particle domain. Then, these two domains are adjusted to be contained in the same computation domain. The Green function is modified to take into account the center shift of field domain. From the shifted Green function, we can compute the electrical potential in the shifted field domain using the FFT based method. This algorithm saves the requirement to contain the original particle domain and the field domain in one big computation domain and is therefore more computationally efficient. To test this algorithm, we calculated the radial electrical field from 0 to $2R_0$ generated by a round beam with Gaussian density distribution and radius R_0 . In this example, the particle domain is defined from $-R_0$ to R_0 for x and y , and the field domain is defined from 0 to $2R_0$. Fig. 1 shows a comparison of the radial electrical field E_r in the field domain as a function of radius from analytical solution and from above shifted Green function algorithm. The two solutions agree with each other very well.

In the beam-beam interaction, the two beams can collide with a crossing angle. A transformation is used to change the crossing angle collision into a head-on collision. The

transformation is given by [4]

$$\begin{aligned} x^* &= x(1 + h_x^* \cos(\alpha) \sin(\psi)) + y h_x^* \sin(\alpha) \sin(\psi) \\ &\quad + z \cos(\alpha) \tan(\psi) \end{aligned} \quad (23)$$

$$\begin{aligned} y^* &= x h_y^* \cos(\alpha) \sin(\psi) + y(1 + h_y^* \sin(\alpha) \sin(\psi)) \\ &\quad + z \sin(\alpha) \tan(\psi) \end{aligned} \quad (24)$$

$$\begin{aligned} z^* &= x h_z^* \cos(\alpha) \sin(\psi) + y h_z^* \sin(\alpha) \sin(\psi) \\ &\quad + z / \cos(\psi) \end{aligned} \quad (25)$$

$$p_x^* = p_x / \cos(\psi) - h \cos(\alpha) \tan(\psi) / \cos(\psi) \quad (26)$$

$$p_y^* = p_y / \cos(\psi) - h \sin(\alpha) \tan(\psi) / \cos(\psi) \quad (27)$$

$$\begin{aligned} p_z^* &= p_z - p_x \cos(\alpha) \tan(\psi) - p_y \sin(\alpha) \tan(\psi) \\ &\quad + h \tan^2(\psi) \end{aligned} \quad (28)$$

where α is the crossing plane angle in the $x - y$ plane, ψ is the half crossing angle in the $\tilde{x} - z$ plane, $h = p_z + 1 - \sqrt{(p_z + 1)^2 - p_x^2 - p_y^2}$, $h_i^* = \frac{\partial}{\partial p_i^*} h^*(p_x^*, p_y^*, p_z^*)$, and $h^*(p_x^*, p_y^*, p_z^*) = h(p_x^*, p_y^*, p_z^*)$.

3 PARALLEL IMPLEMENTATION

We have employed a one-dimensional domain-decomposition approach in the parallel implementation of beam-beam simulation [12]. The physical domain is decomposed along the y direction into a number of small rectangular blocks. These blocks are mapped to a logical one-dimensional Cartesian processor grid. Each processor contains one rectangular block domain. The particles with spatial positions within the local computational boundary are assigned to the processor containing that part of the physical domain. In the multiple slice model, each beam is divided into a number of slices along z direction. Within each slice, a mesh grid is defined to store the field-related quantities such as charge density and electric field. A guard grid outside each side of the physical boundary is used as temporary storage of grid quantities from the neighboring processors. The boundary grid is defined as the outer-most grids inside the physical boundary.

The particles generated on each processor are advanced using the one-term lattice map and damping and quantum excitation map. At the interaction point, during the collision, if a particle moves outside the local computational domain, it is sent to the corresponding processor where it is located. A particle manager function, implemented using the message passing interface (MPI), is defined to handle the particle movement and the inter-processor communication. The particle manager first checks the y position of every particle on each processor. If this position is outside the local computational domain, the particle is copied to one of its two buffers and sent to one of its two neighboring processors. After a processor receives particles from its neighbors, it determines whether some of them need to be further sent out or not. The outgoing particles are counted and copied into two temporary arrays. The remaining particles are copied into another temporary array. This process is repeated until there is no outgoing particle found on all

processors. Then, the particles in the temporary storage, along with the particles left in the original particle array, are copied into a new particle array.

After each particle moves to its local computational domain, a linear cloud-in-cell (CIC) particle-deposition scheme is done for all processors to obtain the charge density on the grid. For the particles located between the boundary grid and computational domain boundary, these particles will also contribute to the charge density on the boundary grids of neighboring processors. Hence, explicit communication is required to send the charge density on the guard grids, which is from the local particle deposition, to the boundary grids of neighboring processors to sum up the total charge density on the boundary grids. With the charge density on the grids, the Poisson equation is solved using a FFT based algorithm described in last section for the open boundary condition. A parallel FFT can be done along x direction simultaneously on all processors. To do FFT along y direction, we have used a global all-to-all communication to transpose distributed y component onto local processor. Then FFT can be done along y simultaneously on all processor. During the inverse FFT, a reverse process is employed to obtain the potential on the original grids.

From the potential on the grid, we calculate the electric field on the grid using a central finite difference scheme. To calculate the electric field on a boundary grid, the potential on a boundary grid of neighboring processors is required. A communication pattern similar to that employed in the charge density summation on the boundary grids is used to send the potential from the boundary grids to the guard grids of neighboring processors. After the electric field on the grids is obtained, it has to be interpolated from the grids onto the local particles to advance the particles. Since we have used the linear CIC scheme, the electric field of particles between the boundary grid and computational domain boundary will also depend on the electric field on the boundary grid of neighboring processors. A similar communication pattern is used to send the electric field from the boundary grids to the guard grids of the neighboring processors. With the electric field on grids local to each processor, the interpolation is done for all processors to obtain the electromagnetic force on every particle from the opposite moving beam. The local particles are updated in momentum space using the beam-beam force.

Dynamic load balancing is employed with adjustable frequency to keep the number of particles on each processor approximately equal. To determine the local boundary, the two-dimensional density function is summed up along the x direction to get the local one-dimensional charge density function along y . The local charge density function is gathered along y direction to get a global y direction charge density distribution function on each processor. Using this global y direction density distribution, the local computational boundary in the y direction can be determined assuming that each processor contains a fraction of the total number of particles about equal to $1/n_{proc}$. Here, n_{proc} is

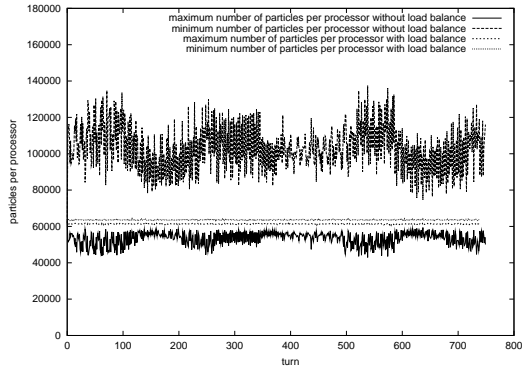


Figure 2: Maximum and minimum number of particles per processor with and without load balance.

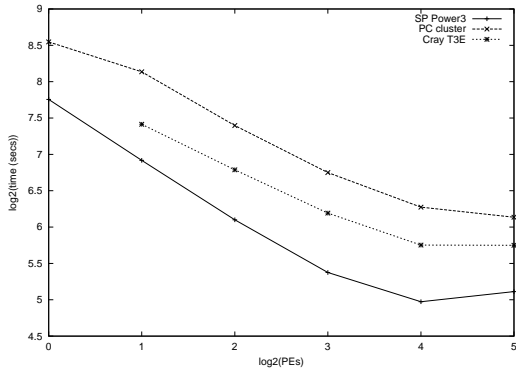


Figure 3: The time cost for 10 turns as a function of processors on IMB SP Power3, Cray T3E, and PC cluster.

the number of processors along the y direction. Given the new local computational boundary, the particles are moved around among processors to their local physical domains. Fig. 2 shows the maximum and minimum number of particles per processor with and without dynamic load balance. With dynamic load balancing, the difference between the maximum number of particles and the minimum number of particles has been drastically reduced. As a preliminary study results, Fig. 3 shows the execution time for 10 turns as a function of the processor number on a SP Power3, a Cray T3E-900 and a PC cluster for a fixed problem size. Here, we have used 1 million particles for each beam. Each beam is divided into 5 slices. For each slice, we have used 64 by 64 grid points. A reasonable scalability is achieved only up to 16 processors. Beyond 16 processors, the execution time starts to saturate and even increase. The degradation of performance with large number of processors is due to the relatively small grid number (64) used in this example. A significant fraction of computation time has been spent on the communication among processors. The small number of grids per processor when 32 processors were used also results in a poor load balance for a Gaussian density distribution which has most particles located around the center of beam. For the given number of processor, the SP Power3 costs the least time compared with two other machines. This is because the peak performance

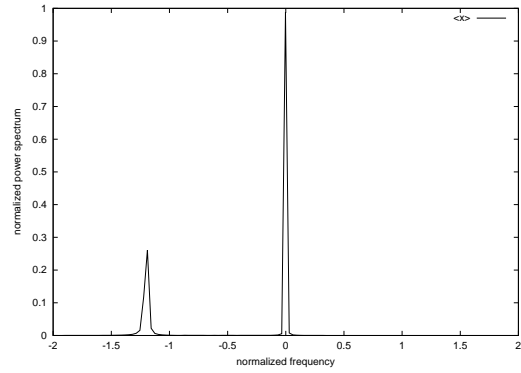


Figure 4: The power spectrum of the x centroid of one beam.

for the SP processor is 1.5 GFlops, and for the T3E is 900 MFlops. The PC cluster uses Pentium III processor with 866 MHz clock speed.

4 APPLICATIONS

As an application, we have studied coherent beam-beam dipole oscillation in the proposed Large Hadron Collider (LHC). Two beams have the same physical parameters moving in an opposite direction. The kinetic energy of each beam is 7 TeV. The horizontal and vertical bare tunes are 0.31. The beam-beam parameter ξ is -0.0034 . Here ξ is defined as $\frac{r_p N \beta_{\perp}}{4 \pi \gamma \sigma_{\perp}^2}$ and r_p is classical proton radius. Fig. 4 shows the power spectrum of the x centroid motion (same for y since it is assumed a round beam) in one beam. The normalized frequency is defined as $(\nu - \nu_{x0})/|\xi|$. It is seen that there exist two oscillation frequencies for the centroid motion. One corresponds to the bare tune without beam-beam interaction. This mode is called σ mode. The oscillation of this mode is in phase for two beams. The other frequency is shifted towards the lower frequency by $(1.19 \pm 0.03)|\xi|$. This mode is called π mode. The oscillation of this mode is 180 degrees out of phase for two beams. The frequency downshift has also been calculated using a linearized Vlasov equation which gives 1.21ξ [13]. We see that the simulation result agrees with the theoretical calculation very well. Fig. 5 shows the RMS emittance as a function of turns in the accelerator. No significant emittance growth of the beam is observed after 9000 turns. In this case, the coherent beam-beam tune shift is very weak and will not cause any significant resonance crossing.

5 SUMMARY

In this paper, we have presented a parallel simulation model to study beam-beam interaction in high energy accelerators. The electromagnetic fields between two colliding beams are calculated using a parallel particle-in-cell approach. Using high performance computers, we can study the physical processes such as long range off-centroid collision, finite bunch length effects, crossing angle collision

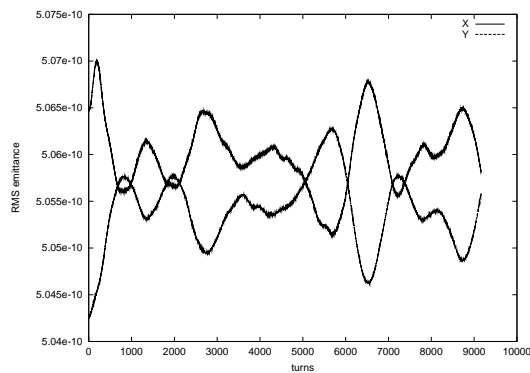


Figure 5: The x and y emittances as a function of turns in the accelerator.

in the beam-beam interaction which can not be done on a serial computer. A preliminary performance test shows a reasonable scalability up to 16 processors for a relatively small problem size. A further performance study will be carried out to improve the scalability of the code and will be presented in the final version of paper. Dynamical load balance has been implemented to reduce the difference of particle distribution among processors. As an application, we have studied the coherent beam-beam oscillation in the proposed LHC. The mode frequency shift from our simulation agrees very well with the theoretical analysis.

6 ACKNOWLEDGMENTS

This work was performed on the Cray T3E and IBM SP at the National Energy Research Scientific Computing Center located at Lawrence Berkeley National Laboratory. This work was supported by the U.S. Department of Energy, Office of Science, Division of High Energy and Nuclear Physics, under the project, Advanced Computing for 21st Century Accelerator Science and Technology.

7 REFERENCES

- [1] K. Hirata, H. Moshhammer and F. Ruggiero, Particle Accelerators **40**, 205 (1993).
- [2] K. Hirata, Phys. Rev. Lett. **20**, 2228 (1995).
- [3] Y. Papaphilippou and F. Zimmermann, PRST-AB **2**, 104001 (1999).
- [4] L. H. A. Leunissen, F. Schmidt, G. Ripken, PRST-AB **3**, 124002 (2000).
- [5] M. A. Furman, A. Zholents, T. Chen, and D. Shatilov, "Comparisons of Beam-Beam Code Simulations," CBP Tech Note-59, 1996.
- [6] S. Krishnagopal and M. A. Furman and W. C. Turner, "Studies of the Beam-Beam Interaction for the LHC," LBNL-43061, CBP Note 308 (1999).
- [7] T. Koyama, PRST-AB **2**, 024001 (1999).
- [8] M. A. Furman, "Beam-Beam Simulations with the Gaussian Code TRS", LBNL-42669, CBP Note 272 (1999).

- [9] M. P. Zorzano and F. Zimmermann, PRST-AB **3**, 044401 (2000).
- [10] M. A. Furman and W. C. Turner, "Beam-Beam Simulations for Separated Beams in the LHC", LBNL-46223, CBP Note 350 (2000).
- [11] R. W. Hockney and J. E. Eastwood, "Computer Simulation Using Particles," McGraw-Hill Book Company, New York, 1985.
- [12] P. C. Liewer and V. K. Decyk, J. Comput. Phys. **85**, 302 (1989).
- [13] K. Yokoya and H. Koiso, Part. Accel. **27**, 181 (1990).